Suunnitteludokumentti

Boa Open Access

Helsinki 3.5.2006

Ohjelmistotuotantoprojekti

HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

Kurssi

581260 Ohjelmistotuotantoprojekti (6 ov)

Projektiryhmä

Ilmari Heikkinen
Timo Hintsa
Erno Härkönen
Arto Vuori
Mikko Kautto

Asiakas

Olli Niinivaara

Johtoryhmä

Juha Taina
Riikka Kaven

Kotisivu

`http://www.cs.helsinki.fi/group/boa`

Versiohistoria

Versio	Päiväys	Tehdyt muutokset
1.0	20.3.2006	Ensimmäinen versio
2.0	3.4.2006	Toisen iteraation aiheuttamia muutoksia
2.5	7.4.2006	Korjauksia ja lisäyksiä
2.7	11.4.2006	Ulkoasua hiottu
2.8	14.4.2006	Korjauksia
2.9	3.5.2006	Korjauksia

Sisältö

1 Johdanto

Tämä on Boa Open Access -ryhmän kevään 2006 ohjelmistotuotantoprojektikurssilla toteuttaman Open Access -ohjelmiston suunnitteludokumentti. Suunnitteludokumentissa kuvataan ohjelmiston yleisarkkitehtuuri, osajärjestelmien väliset rajapinnat ja osajärjestelmät tilanteen vaatimalla tarkkuudella. Osajärjestelmien sisäistä toimintaa tietorakenteineen ja algoritmeineen on selitetty tarkemmin, kun se on ollut hyödyllistä.
Suunnitteludokumenttia on kirjoitettu iteratiivisesti, koska projekti toteutetaan iteratiivisella tuotantomenetelmällä.

2 Sanasto

Qriterium Metadataympäristö [6]

Citeseer Open Access -tietokanta [2]

DBLP Digital Bibliography and Library Project. Tietojenkäsittelytiedettä käsittelevien artikkeleiden tietokanta. Huhtikuussa 2006 DBLP listasi yli 740 000 artikkelia. DBLP on perustettu 1980-luvulla. [3]

Transformaattori Muuntaja, muuttaja. Transformaattorin tehtävä on yhtenäistää erilailla organisoitua metadataa yhtenäiseen muotoon.

Qs-lause Qriterium Statement. Qriterium-ympäristön metadatankuvausmenetelmä, johon erilailla organisoitu metadata muunnetaan. XML-muotoista.

QStatement Kuvaa yhtä Qs-lausetta. QStatement voi kuvata esimerkiksi dokumenttia, henkilöä, organisaatiota tai tietolähdettä.

3 Yleisarkkitehtuuri

Ohjelma koostuu kuudesta pääkomponentista ja ulkoisista komponenteista, joista kerrotaan luvussa "Ulkoiset komponentit".

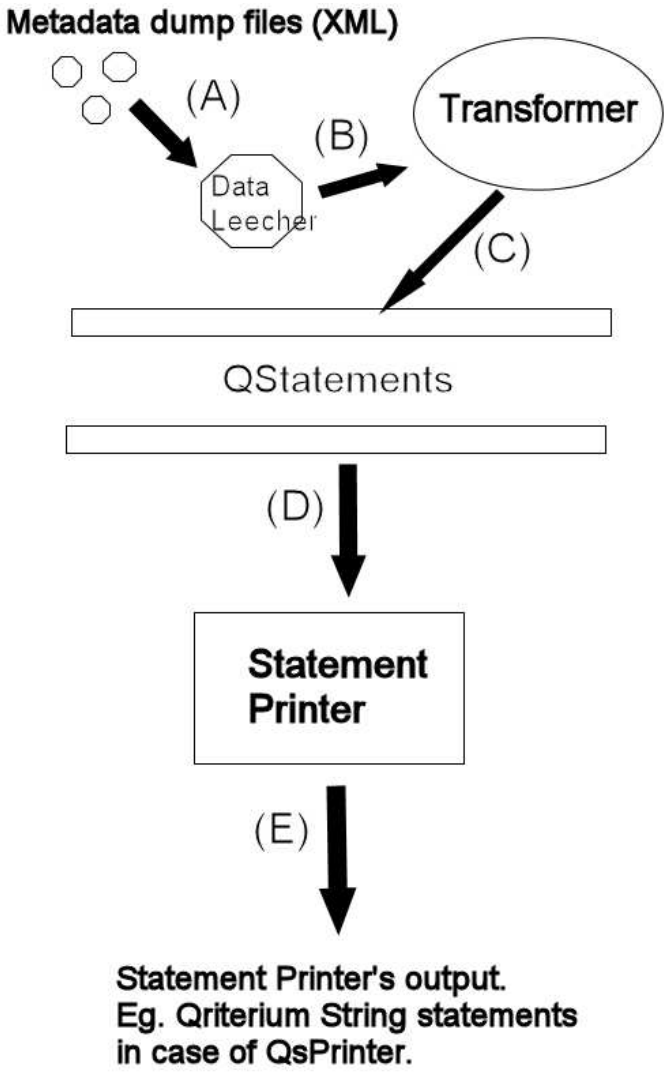
Tiedostonlukija Lukee levyltä dataa ja tarjoaa sen eteenpäin.

Citeseer-transformaattori Muuntaa Citeseerin metadataa QStatementeiksi.

DBLP-transformaattori Muuntaa DBLP:n metadataa QStatementeiksi.

Qs-transformaattori Lukee Qs-lauseita järjestelmään. Muuntaa lauseet takaisin QStatementeiksi.

QStatements-komponentti Vastaanottaa transformaattorilta QStatementteja ja pitää niiden keskinäiset yhteydet ehjinä. Luovuttaa QStatementit edelleen Statement-tulostajille.



Kuva 1: Ohjelman yleisarkkitehtuuri

Statement-tulostaja Tulostaa QStatementteja. Esimerkiksi Qs-tulostaja, joka on yksi Statement-tulostajan erikoistapaus, tulostaa Qs-lauseita joko pakkaamattomana tai gzip-pakattuna. Se voi pilkkoa tulosteen halutun kokoisiksi palasiksi.

Kuvassa 1 nuolet kuvaavat datan kulkusuuntaa. DataLeecher lukee metadataa jostain lähteestä, esimerkiksi levyltä, (A) ja luovuttaa datan transformaattorille InputStreamina (B). Transformaattori luo QStatementteja saadusta datasta ja antaa niitä QStatements-komponentille (C). QStatements-komponenttia kuuntelevat statement-tulostajat saavat tiedon uusista QStatementeista reaaliajassa (D) ja tulostavat QStatementit oman toteutuksensa mukaisesti. Esimerksiksi Qs-tulostajan tulosteena saadaan (E) Qs-lauseita (XML-tiedostoja).

4 Pääkomponentit

Tässä osassa on pyritty valottamaan komponenttien tehtäviä tarkemmin. Komponentit on jaettu luokkiin ja rajapintaluokkiin (Javan "interface"), jotka esitellään tässä osassa. Luokkien rajapinnat kuvataan tarkemmin seuraavassa luvussa.

4.1 Tiedostonlukija

Tässä ohjelmassa metadata-tiedostot haetaan levyltä, eikä esimerkiksi OAI-PMH-protokollalla suoraan internetistä. Tiedostonlukija-komponentti huolehtii datan saamisesta eteenpäin transformaattoreille.

Metadataa on saatavilla mitä erilaisimmissa formaateissa ja sitä voidaan saattaa transformaattoreille erilaisilla tavoilla erilaisten protokollien yli. Tiedostonlukija kapsuloi datan hankintakeinot ja tarjoaa datan eteenpäin InputStreamina. Transformaattoreiden näkökulmasta jostain tulee vain datavirtaa, eikä niiden tarvitse erotella sitä, onko data alunperin tullut levyltä tiedostosta vai jollain muulla tavalla.

4.1.1 Luokat ja rajapintaluokat

DataLeecher Rajapinta, jonka tiedostonlukija toteuttaa. Toteuttajat kapsuloivat sen, miten ja mistä data on hankittu, ja tarjoavat transformaattorille InputStreamin.

FileReader Tiedostonlukija-luokka, joka toteuttaa DataLeecher-rajapinnan.

4.1.2 Metodien paluuarvot

Luokka	Metodi	paluuarvon selite
DataLeecher	abstract getInputStream()	*InputStream*
DataLeecher	abstract getFileName()	*String*, luettavan tiedoston nimi
FileReader	getInputStream()	*InputStream*
FileReader	getFileName()	*String*

4.2 Citeseer-transformaattori

Citeseer-transformaattori erottelee saamastaan XML-datasta tietueet, jotka on merkitty citeseer-datassa record-tagein. Olemme huomanneet, että Citeseerin tarjoamassa datassa kaikki recordit eivät ole validia XML:ää. Vialliset recordit hylätään. Kun viallinen record osuu kohdalle, siirrytään seuraavaan.

XML:n jäsentämisessä käytetään XPathia (DOM-pohjainen jäsentäjä).

Transformaattoreiden pitää pystyä erottelemaan syötteestään kiinnostava data. Qriterium Statement (ks. Vaatimusdokumentti [1]) määrää, mikä on kiinnostavaa dataa.

Transformaattori luo oikean tyyppisiä QStatement-instansseja jäsentämistään recordeista. Oikealla tyypillä tarkoitetaan sitä, että transformaattorin täytyy tunnistaa, mitä recordissa kuvataan. Kuvauksen kohteena voi olla itse dokumentin lisäksi muuttuva määrä tekijöitä. Tekijät voivat taas olla ihmisiä tai organisaatioita. Dokumenttien välillä on yhteyksiä. Yhteydet voivat olla esimerkiksi dokumenttien välisiä. Myös muunlaisia yhteyksiä on. Dokumentin ja sen tekijän välillä on aina yhteys. Transformaattorin on kyettävä tunnistamaan, minkä tyyppinen yhteys on kyseessä ja luoda vastaavan tyyppisiä QStatement-instansseja.

4.2.1 Luokat ja rajapintaluokat

Transformer Abstrakti yläluokka kaikille transformaattoreille

CiteseerTransformer Citeseer-metadataa ymmärtävä transformaattori.

4.2.2 Metodien paluuarvot

Luokka	Metodi	paluuarvon selite
Transformer	transform(DataLeecher, QStatements)	*void*, suorittaa muuntamisen
Transformer	getValidItems()	*long*, ehjien muunnosten määrä
Transformer	getInvalidItems()	*long*, epäonnistuneiden muunnosten määrä
CiteseerTransformer	transform(DataLeecher, QStatements)	*void*, suorittaa muuntamisen
CiteseerTransformer	getValidItems()	*long*, ehjien muunnosten määrä
CiteseerTransformer	getInvalidItems()	*long*, epäonnistuneiden muunnosten määrä

4.3 DBLP-transformaattori

DBLP-transformaattori tuo DBLP:n tarjoamia artikkeleita järjestelmään. Transformaattori toteutetaan hyvin samalla tavalla kuin Citeseer-transformaattori. Sisään luettu data on eri muodossa kuin Citeseer-data, mutta tämänkin transformaattorin tarkoituksena on luoda syötteestään QStatement-instansseja.
DBLP-transformaattori on yleisen Transformer-luokan erikoistapaus.

4.3.1 Luokat ja rajapintaluokat

DBLPTransformer DBLP-metadataa ymmärtävä transformaattori.

4.3.2 Metodien paluuarvot

Luokka	Metodi	paluuarvon selite
DBLPTransformer	transform(DataLeecher, QStatements)	*void*, suorittaa muuntamisen
DBLPTransformer	getValidItems()	*long*, ehjien muunnosten määrä
DBLPTransformer	getInvalidItems()	*long*, epäonnistuneiden muunnosten määrä

4.4 Qs-transformaattori

Qs-transformaattori osaa muuntaa Qs-tulostajan tuottamia Qs-lauseita takaisin QStatementeiksi. Qs-transformaattori on yleisen Transformer-luokan erikoistapaus.

4.4.1 Luokat ja rajapintaluokat

QsTransformer Qs-lauseita ymmärtävä transformaattori.

4.4.2 Metodien paluuarvot

Luokka	Metodi	paluuarvon selite
QsTransformer	transform(DataLeecher, QStatements)	*void*, suorittaa muuntamisen
QsTransformer	getValidItems()	*long*, ehjien muunnosten määrä
QsTransformer	getInvalidItems()	*long*, epäonnistuneiden muunnosten määrä

4.5 QStatements-komponentti

QStatements vastaanottaa transformaattorilta QStatementteja ja pitää niiden keskinäiset yhteydet ehjinä. QStatements-komponenttia kuuntelevat statement-tulostajat saavat QStatements-komponentilta tiedon uusista QStatementeista reaaliajassa.

4.5.1 Luokat ja rajapintaluokat

QStatements Vastaanottaa QStatementteja ja ilmoittaa niistä kuunteleville statement-tulostajille.

QStatement Kuvaa yhtä Qriterium-statementtia.

QHeader Kuvaa Qriterium-statementin otsaketta.

QDocument QStatementin ilmentymä. Kuvaa dokumentin metadataa.

QActor QStatementin ilmentymä. Kuvaa henkilöä tai organisaatiota. Jakautuu alatyyppeihin Person ja Organization.

Person Kuvaa henkilöä.

Organization Kuvaa organisaatiota.

QConnection QStatementin ilmentymä. Kuvaa yhteyttä kahden Qriterium-statementin välillä. QConnectionit on edelleen jaettu alatyyppeihin (CreatedForConnection, ContactedAtConnection, PublishedByConnection, StoredByConnection, SupportedByConnection, CreatedInConnection, ManagedByConnection, IdentifiedAsConnection, ConsistsOfConnection, AwareOfConnection, CreatedByConnection).

QSource QStatementin ilmentymä. Kuvaa "raakadatan"lähdettä. Raakadataa on esimerkiksi Citeseerin metadatadumpit.

QContent Qstatementin ilmentymä. Kuvaa metadataa dokumentin sisällöstä.

QStatementIndex Indeksoi Boan sisäiset tunnisteet käyttäen avaimena raa'an metadatasyötteen omia tunnisteita. QStatementIndexiltä voidaan myöhemmin selvittää, onko joku QStatement jo ennestään tuttu järjestelmälle.

QConnectionHandler Luo QConnectioneja sitä mukaan, kun se on mahdollista. QConnectionia ei voi luoda kahden QStatementin välille, ennen kuin kummatkin osapuolet on luettu sisään ohjelmaan. Säilyttää tilansa eräajojen välillä toteuttamansa Storable-rajapinnan avulla.

RawConnection QConnectionHandlerin tarvitsema apuluokka. Transformaattorit eivät luo suoraan QConnectioneja vaan RawConnectioneja, koska Transformaattori ei tiedä, tuleeko yhteyden toista osapuolta löytymään ikinä tulevaisuudessa. QConnectionHandlerin vastuulle jää oikeiden QConnectionien luominen saamiensa RawConnection-olioiden ja QStatements-komponentin antamien signaalien perusteella.

QConnectionFactory Luo uusia halutun tyyppisiä QConnectioneja tyyppimerkkijonon perusteella.

IdFactory Huolehtii QStatementtejen sisäisten tunnisteiden yksilöllisyydestä. Luovuttaa pyydettäessä uniikin id:n. Säilyttää tilansa eräajojen välillä toteuttamansa Storable-rajapinnan avulla.

XMLDataElement QStatementin sisäiset elementit, eli avain-arvo-parit on saatavissa ulos XMLDataElement-muodossa.

4.5.2 Metodien paluuarvot

Luokka	Metodi	paluuarvon selite
QStatements	addQStatement(QStatement)	*boolean*, true jos QStatement välitettiin statement-tulostajille, muutoin false.
QStatements	addQStatement(RawConnection)	*boolean*, true jos QStatement välitettiin statement-tulostajille.
QStatements	getQConnectionHandler()	*QConnectionHandler*, testausta varten.
QStatements	getStatementIdByRawId(String)	*Long|null*, ohjelman sisäinen QStatement-tunniste Long-oliona, jos indeksi löytää QStatementin annetulla rawId:llä. Muutoin palautetaan null.
QStatements	getIndex()	*QStatementIndex*, testausta varten.
QStatements	addPrinter(StatementPrinter)	*void*, lisää statement-tulostajan QStatements-komponentin kuuntelijaksi.
QStatement	setHeader(QHeader)	*void*, asettaa headerin
QStatement	getHeader()	*QHeader*, palauttaa headerin
QStatement	getId()	*long*, QStatementin sisäinen tunniste
QStatement	getHash()	*String*, alkuperäisen datalähteen tunnistearvo
QStatement	getRawId()	*String*, alkuperäisen datalähteen antama tunniste
QStatement	getAt()	*Date*, QStatementin luontipäivämäärä
QStatement	isConnectable()	*boolean*, true jos QStatement on yhdistettävissä QConnectionilla, muutoin false.
QStatement	getXMLNamespace()	*String*, XML-nimiavaruus
QStatement	toString()	*String*, merkkijonoesitysmuoto
QStatement	abstract getXMLDataElements()	*XMLDataElement[]*, XML-esitysmuodon avain-arvoparit
QStatement	abstract getStartTag()	*String*, XML-esityksen vaatima aloitustagi
QStatement	abstract getEndTag()	*String*, XML-esityksen vaatima lopetustagi
QHeader	getId()	*long*, QStatementin sisäinen tunniste
QHeader	getHash()	*String*, alkuperäisen datalähteen tunnistearvo
QHeader	getRawId()	*String*, alkuperäisen datalähteen antama tunniste
QHeader	getAt()	*Date*, QStatementin luontipäivämäärä
QHeader	getXMLDataElements()	*XMLDataElement[]*, XML-esitysmuodon avain-arvoparit

Luokka	Metodi	paluuarvon selite
QDocument	setTitle(String)	*void*
QDocument	setLanguage(String)	*void*
QDocument	setFilename(String)	*void*
QDocument	getTitle()	*String*, dokumentin otsikko
QDocument	getLanguage()	*String*, dokumentin kieli
QDocument	getFilename()	*String*, dokumentin sijainti (URI)
QDocument	getStartTag()	*String*, XML-esityksen vaatima aloitustagi
QDocument	getEndTag()	*String*, XML-esityksen vaatima lopetustagi
QDocument	getXMLDataElements()	*XMLDataElement[]*, XML-esitysmuodon avain-arvoparit
QActor	setName(String)	*void*
QActor	getName()	*String*, actorin nimi
QActor	isConnectable()	*boolean*, aina true, koska QActorit on aina yhdistettävissä QConnectionilla
QActor	getStartTag()	*String*, XML-esityksen vaatima aloitustagi
QActor	getEndTag()	*String*, XML-esityksen vaatiuma lopetustagi
QActor	getXMLDataElements()	*XMLDataElement[]*, XML-esitysmuodon avain-arvoparit
QActor	abstract getSubTypeString()	*String*, actorin alatyyppi
Person	getSubTypeString()	*String*, henkilön oma alatyyppitunniste
Organization	getSubTypeString()	*String*, organisaation oma alatyyppitunniste
QConnection	setFrom(long)	*void*
QConnection	setFrom(Long)	*void*
QConnection	setTo(long)	*void*
QConnection	setTo(Long)	*void*
QConnection	getFrom()	*Long*, yhteyden viittaavan pään sisäinen tunniste
QConnection	getTo()	*Long*, yhteyden viitattavan pään sisäinen tunniste
QConnection	getStartTag()	*String*, XML-esityksen vaatima aloitustagi
QConnection	getEndTag()	*String*, XML-esityksen vaatima lopetustagi
QConnection	getXMLDataElements()	*XMLDataElement[]*, XML-esitysmuodon avain-arvoparit
QConnection	toString()	*String*, merkkijonoesitysmuoto yhteydestä
QConnection	abstract getTypeString()	*String*, yhteyden tunnistemerkkijono

Luokka	Metodi	paluuarvon selite
AwareOfConnection	getTypeString()	*String*, yhteyden tunnistemerkkijono
ConsistOfConnection	getTypeString()	*String*, yhteyden tunnistemerkkijono
ContactedAtConnection	getTypeString()	*String*, yhteyden tunnistemerkkijono
CreatedByConnection	getTypeString()	*String*, yhteyden tunnistemerkkijono
CreatedForConnection	getTypeString()	*String*, yhteyden tunnistemerkkijono
CreatedInConnection	getTypeString()	*String*, yhteyden tunnistemerkkijono
IdentifiedAsConnection	getTypeString()	*String*, yhteyden tunnistemerkkijono
ManagedByConnection	getTypeString()	*String*, yhteyden tunnistemerkkijono
PublishedByConnection	getTypeString()	*String*, yhteyden tunnistemerkkijono
StoredByConnection	getTypeString()	*String*, yhteyden tunnistemerkkijono
SupportedByConnection	getTypeString()	*String*, yhteyden tunnistemerkkijono
QSource	setName(String)	*void*
QSource	setOpenURL(URL)	*void*
QSource	getName()	*String*
QSource	getOpenURL()	*URL*
QSource	setEncoding(String)	*void*
QSource	getEncoding()	*String*
QSource	setFormat(String)	*void*
QSource	getFormat()	*String*
QSource	setTerminology(String)	*void*
QSource	getTerminology()	*String*
QSource	setCompression(String)	*void*
QSource	getCompression()	*String*
QSource	setBackprotocol(String)	*void*
QSource	getBackProtocol()	*String*
QSource	setToprotocol(String)	*void*
QSource	getToprotocol()	*String*
QSource	setLaunched(Date)	*void*
QSource	getLauched()	*Date*
QSource	getStartTag()	*String*, XML-esityksen vaatima aloitustagi
QSource	getEndTag()	*String*, XML-esityksen vaatiuma lopetustagi
QSource	getXMLDataElements()	*XMLDataElement[]*, XML-esitysmuodon avain-arvoparit
QContent	isConnectable()	*boolean*, aina true, koska QContent on yhdistettävissä QConnectionilla
QContent	toString()	*String*, merkkijonoesitysmuoto QContentista
QContent	getStartTag()	*String*, XML-esityksen vaatima aloitustagi
QContent	getEndTag()	*String*, XML-esityksen vaatiuma lopetustagi
QContent	getXMLDataElements()	*XMLDataElement[]*, XML-esitysmuodon avain-arvoparit

Luokka	Metodi	paluuarvon selite
QStatementIndex	put(QStatement)	*void*, lisää QStatementin indeksiin
QStatementIndex	putRaw(String s, Long l)	*void*, lisää indeksiin alkion, jonka avain on s ja arvo on l
QStatementIndex	get(String s)	*Long|null*, palauttaa Long-arvon, jos avaimella s on alkio indeksissä. Jos mitään ei löydy, palauttaa null
QStatementIndex	store()	*boolean*, true jos QStatementIndex'n tilan tallennus onnistui, muutoin false.
QStatementIndex	store(File)	*boolean*, sama kuin yllä, mutta tallennustiedosto on määriteltävissä. Metodi on testejä varten tehty.
QStatementIndex	retrieve()	*boolean*, true jos QStatementIndex'n tilan palautus onnistui, muutoin false.
QStatementIndex	retrieve(File)	*boolean*, sama kuin yllä, mutta palautustiedosto on määriteltävissä. Metodi on testejä varten tehty.
QConnectionHandler	addRawConnection(RawConnection)	*void*, lisää uuden RawConnectionin QConnectionHandleriin
QConnectionHandler	statementAdded(String))	*int*, luotujen QConnection-olioiden määrä
QConnectionHandler	store()	*boolean*, true jos QConnectionHandler'n tilan tallennus onnistui, muutoin false.
QConnectionHandler	store(File)	*boolean*, sama kuin yllä, mutta tallennustiedosto on määriteltävissä. Metodi on testejä varten tehty.
QConnectionHandler	retrieve()	*boolean*, true jos QConnectionHandler'n tilan palautus onnistui, muutoin false.
QConnectionHandler	retrieve(File)	*boolean*, sama kuin yllä, mutta palautustiedosto on määriteltävissä. Metodi on testejä varten tehty.
QConnectionHandler	getHashSize()	*int*, RawConnection-olioiden määrä QConnectionHandlerissä
QConnectionHandler	getRawConnections()	*RawConnection[]*, palauttaa RawConnectionit taulukossa. Tämä metodi on testejä varten olemassa.
RawConnection	getTo()	*String*, viittaavan statementin rawId
RawConnection	getFrom()	*String*, viitattavan statementin rawId
RawConnection	getType()	*String*, yhteyden tyypin merkkijonoesitysmuoto
RawConnection	toString()	*String*, merkkijonoesitysmuoto
RawConnection	getStartTag()	*String*, XML-esityksen vaatima aloitustagi
RawConnection	getEndTag()	*String*, XML-esityksen vaatima lopetustagi
RawConnection	getXMLDataElements()	*XMLDataElement[]*, XML-esitysmuodon avain-arvoparit
QConnectionFactory	getInstance(String type)	*QConnection|null*, jos type on validi QConnection-tyyppi palauttaa pyydetyn tyyppisen QConnection-olion. Jos tyyppi on tuntematon palautetaan null.

Luokka	Metodi	paluuarvon selite
IdFactory	getUniqueId	*long*, uniikki tunniste QStatement-oliota varten
IdFactory	store()	*boolean*, true jos IdFactory'n tilan tallennus onnistui, muutoin false.
IdFactory	retrieve()	*boolean*, true jos IdFactory'n tilan palautus onnistui, muutoin false.
XMLDataElement	getNamespace()	*String*, XML-elementin nimiavaruus
XMLDataElement	getName()	*String*, XML-elementin nimi
XMLDataElement	getValue()	*String*, XML-elementin arvo

4.5.3 QConnectionHandler

QStatements-komponentin QConnectionHandlerin toimintaa on syytä selventää lisää.

Tehtävät

- Luo QConnection-instansseja.
- Ylläpitää vaillinaisia QConnectioneja (RawConnectioneja)

Toteutus QConnectionHandler:lle annetaan RawConnection-olioita, jotka sisältävät kahden QDocumentin rawId:t. Handler ylläpitää RawConnection-olioita TreeMap-tietorakenteessa (Sunin toteutus punamustasta puusta). TreeMapiin on päädytty, koska RawConnectioneja pitää pystyä jatkuvasti etsimään, lisäämään ja poistamaan tietorakenteesta tehokkaasti. Punamusta puu takaa takaa muokkausoperaatioille ajan O(n log n) ja etsimiselle aikaluokan O(log n), jossa n on solmujen määrä puussa. Puun avaimena on RawConnectionista saatu toRawId (String). QStatements-olio antaa uuden RawConnectionin Handlerille aina kun transformaattori sellaisen sille luovuttaa. Transformaattorin pitää antaa RawConnection-instanssille QHeader, josta myöhemmin kaivetaan osa QConnectionin otsaketiedoista, ja QConnectionin tyypi (String).

Kun transformaattori löytää syötteestään uuden statementin, tieto siitä tulee QStatements-olion kautta Handlerille metodilla statementAdded(QStatement statement). Handleri etsii RawConnection-olioita puurakenteestaan käyttäen syötetyn QStatementin rawId:tä. Jos RawConnection löytyy, tarkoittaa se sitä, että voidaan luoda uusi QConnection-instanssi. Uusi QConnection saa headertiedoistaan RawConnectionilta seuraavat kentät: qs:rawId, qs:origin, qs:at. Handleri luo oikeantyyppisen QConnection-instanssin QConnectionFactoryn avulla käyttäen RawConnectioniin aikaisemmin säilöttyä tyyppi-stringiä.

Lopulta QConnectionHandler luovuttaa uuden QConnectionin QStatements-komponentille, joka ilmoittaa normaalisti uudesta QStatementista kuunteleville statement-tulostajille. Tätä tehdään, kunnes QConnectionHandlerin RawConnection-puusta ei enää löydy solmuja ko. toRawId:llä.

4.5.4 QStatementIndex

QStatementIndexin tehtävä on pitää kirjaa QStatementeista (sellaisista, joihin on mahdollista liittää yhteys "QConnection"), jotta on mahdollista sekä eräajon aikana että seuraavilla ajokerroilla selvittää, onko jokin uudelleen vastaan tullut QStatement jo ennestään tuttu järjestelmälle, eli onko sillä jo sisäinen tunniste. Mikäli sisäinen tunniste löytyy jo, ei tarvitse luoda uutta QStatementtia vaan käytetään jo ennen luotua.

QStatementtien uutuus selvitetään kysymällä QStatementIndexiltä; löytyykö tietylle rawId:lle jo merkintää. QStatementIndex palauttaa vastauksena sisäisen tunnisteen, tai null jos vastaavuutta ei löydy.

QStatements-olio ilmoittaa QStatementIndexille uusista QStatementeista kutsumalla indeksin put(String rawId, QStatement statement) -metodia. Koska indeksin pitää säilyttää tilansa eräajojen välissä, sen tila pitää olla tallennettavissa levylle. QStatementIndex toteuttaa rajapintaluokan Storable, jonka metodit store() ja retrieve() mahdollistavat halutun toiminnallisuuden. Talletusmuotona käytetään XML:ää.

4.6 Statement-tulostaja

Tämän ohjelman tarkoitus on pääasiassa tuottaa Qriterium String -muotoista XML-ulostuloa - - eli Qs-lauseita. Tässä kappaleessa selvitetäänkin Qs-tulostajan toimintaa.

Qs-tulostajan tehtävä on luoda Qriterium-muotoista XML-dataa (ks. vaatimusdokumentti [1]) QStatements-komponentilta saaduista QStatement-oliosta. Qs-tulostaja luo XML-tiedostoja halutun kokoisissa palasissa. Myös gzip-pakattu ulostulo on mahdollista.

4.6.1 Luokat ja rajapintaluokat

StatementPrinter Rajapinta statement-tulostajille

QSPrinter Tulostaa Qriterium-muotoisia XML-tiedostoja QStatement-olioista.

XMLProperties Luokka, joka tarjoaa Qriterium-XML:n tarvitsemia vakioita.

4.6.2 Metodien paluuarvot

Luokka	Metodi	paluuarvon selite
StatementPrinter	abstract print(QStatement)	*void*, tulostaa parametrinä annetun QStatementin
StatementPrinter	abstract close()	*void*, viimeistelee tulostuksen
QsPrinter	print(QStatement)	*void*, tulostaa QStatementin Qs-muodossa
QSPrinter	close()	*void*, viimeistelee tulostuksen
QsPrinter	getFilePath()	*String*, tulostetiedostojen hakemisto
QsPrinter	setFilePath(String)	*void*, asettaa tulostushakemiston
QsPrinter	getFileName()	*String*, tulostiedostojen nimi (etuliite)
QsPrinter	setFileName(String)	*void*, asettaa tulostiedoston nimen (etuliiteen)
QsPrinter	isGzip()	*boolean*, true jos tulostetaan gzip-pakattua dataa, muutoin false
QsPrinter	setGzip(boolean)	*void*
QsPrinter	getOutputSize()	*long*, tulostustiedostojen maksimikoko tavuina
QsPrinter	setOutputSize(long)	*void*, asettaa tulostustiedostojen maksimikoon tavuina
QsPrinter	getDataWriter()	*DataWriter*, (XML-DataWriter)
QsPrinter	setDataWriter(DataWriter)	*void*, asettaa XML-DataWriter'n
QsPrinter	getOutStreamWriter()	*OutputStreamWriter*
QsPrinter	getCurrentFileName()	*String*, nykyinen tiedostonimi, johon tulostetaan

5 Apukomponentit

Apuluokat, jotka eivät varsinaisesti ole minkään muun komponentin osia, esitellään tässä kappaleessa.

LogFormatter Lokitietojen kirjaaja

StatusPrinter Edistymisilmaisin käyttäjää varten

StringUtilities Apu-string-funktioita

Storable Rajapinta olion tilan säilyttämiseen ja palauttamiseen

Connection Rajapinta yksisuuntaisille yhteyksille

Source Metadataa metadatalähteestä

User Metadataa boa-käyttäjästä

SourceState Kuvaa metadatalähteen tilaa eräajojen välillä ja ylläpitää FileState-olioita

FileState Kuvaa yhden syötetiedoston tilaa eräajoen jälkeen

Loader Apuluokka konfiguraatiotiedostojen lukemiseen

5.1 Metodien paluuarvot

Luokka	Metodi	paluuarvon selite
LogFormatter	format(LogRecord)	*String*, lokitieto
StatusPrinter	printTemp(String)	*void*
StatusPrinter	print(String)	*void*
StatusPrinter	println(String)	*void*
StatusPrinter	printProgressChar()	*void*
StringUtilities	zeroFill(long, int)	*String*, <long> merkkiä pitkä merkkijono mahdollisine etunollineen
Storable	store()	*boolean*, true jos olion tilan tallennus onnistui, muutoin false.
Storable	retrieve()	*boolean*, true jos olion tilan palautus onnistui, muutoin false.
Connection	getTo()	*Object*, yhteyden viittaaja
Connection	getFrom()	*Object*, yhteyden viitattava
Source	getFileName()	*String*
Source	getName()	*String*
Source	getUser()	*String*
Source	getInputDir()	*String*
Source	getOutputDir()	*String*
Source	getDataDir()	*String*
Source	getTransformer()	*String*
Source	getPrinter()	*String*
Source	getOutputFile()	*String*
Source	getGzip()	*String*
Source	getCutSize()	*long*
User	getName()	*String*
SourceState	getActiveState()	*SourceState*
SourceState	isCompleted()	*String*
SourceState	getFileNames()	*void*
SourceState	setFileState(String, FileState)	*void*
SourceState	getFileState(String)	*FileState*
SourceState	setNextFileID(String)	*void*
SourceState	setNextSerialID()	*void*
SourceState	setSourceID(long)	*void*
SourceState	getSourceID()	*long*
SourceState	getNextFileID()	*long*
SourceState	getNextSerialID()	*long*
SourceState	updateLastRun()	*void*
SourceState	getLastRun()	*Date*
SourceState	toString()	*String*
FileState	getCreationID()	*long*
FileState	setCreationID(long)	*void*
FileState	getDate()	*Date*
FileState	setDate(Date)	*void*
FileState	getInvalidItems()	*long*
FileState	setInvalidItes(long)	*void*
FileState	getValidItems()	*long*
FileState	setValidItems(long)	*void*
Loader	loadSettings(String)	*void*
Loader	loadSettings()	*void*

6 Ulkoiset komponentit

Ohjelmassa on hyödynnetty ulkoisia komponentteja tilanteissa, joissa juuri meidän tarkoitukseen sopiva komponentti on ollut vapaasti saatavilla.

XML-Writer XML-tulostaja. Hallitsee nimiavaruudet kätevästi. [4]

JArgs Komentoriviparametrijäsentäjä [5]

7 Rajapinnat ja luokkien keskinäiset suhteet

Yllä kuvattiin luokkien rajapinnat. Luokkien keskinäiset suhteet kuvataan erikseen UML-kaavion avulla. Kaaviota voidaan sanoa tarkaksi, sillä se kattaa miltei koko järjestelmän lukuun ottamatta muutamaa arkkitehtuurin kannalta epäolennaista luokkaa.

Kaavio tulee erikseen liitteenä (*boa_uml.pdf*).

8 Datalähde- ja käyttäjäkonfiguraatiotiedostot

Datalähteille - - esimerkiksi Citeseerille ja DBLP:lle - - luodaan konfiguraatiotiedostot, joissa säilytetään transformaattoreiden tarvitsemaa tietoa. Konfiguraatiotiedostoissa pidetään kirjaa muunnettujen QStatementtien määrästä ko. lähteestä.

Ohjelman käyttäjille luodaan omat konfiguraatiotiedostot. Tiedosto sisältää käyttäjän yksilöivän tunnisteen ja muuta tietoa käyttäjästä, kuten henkilön nimen. Yhtä lähdekonfiguraatiota kohden voi olla yksi käyttäjäkonfiguraatio. Yksi käyttäjäkonfiguraatio voi liittyä moneen lähdekonfiguraatioon.

Konfiguraatiotiedostoja käytetään myös mainostarkoituksessa muunnetun Qs-datan yhteydessä. Ulkopuoliset saavat selville, kuka on tarjonnut tiedon.

9 Pääohjelma

Pääohjelman (org.qriterium.ui.Boa), tehtävänä on rakentaa Citeseer- DBLP- tai Qs-tyyppisestä datasta Qriterium String -muotoista XML-dataa (Qs-lauseita) käyttäen edellä mainittuja komponentteja.

Pääohjelma toimii konsolissa.

Lähteet

1 http://www.cs.helsinki.fi/group/boa/docs/vaatimusdokumentti.pdf

2 http://citeseer.ist.psu.edu/

3 http://www.informatik.uni-trier.de/ ley/db/

4 http://megginson.com/

5 http://jargs.sourceforge.net/

6 http://qriterium.org/